Concept-to-text Generation via Discriminative Reranking

Concept-to-text Generation via Discriminative Reranking

Ioannis Konstas and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
i.konstas@sms.ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

This paper proposes a data-driven method for concept-to-text generation, the task of automatically producing textual output from non-linguistic input. A key insight in our approach is to reduce the tasks of content selection ("what to say") and surface realization ("how to say") into a common parsing problem. We define a probabilistic context-free grammar that describes the structure of the input (a corpus of database records and text describing some of them) and represent it compactly as a weighted hypergraph. The hypergraph structure encodes exponentially many derivations, which we rerank discriminatively using local and global features. We propose a novel decoding algorithm for finding the best scoring derivation and generating in this setting. Experimental evaluation on the ATIS domain shows that our model outperforms a competitive discriminative system both using BLEU and in a judgment elicitation study.

1 Introduction

Concept-to-text generation broadly refers to the task of automatically producing textual output from non-linguistic input such as databases of records, logical form, and expert system knowledge bases (Reiter and Dale, 2000). A variety of concept-to-text generation systems have been engineered over the years, with considerable success (e.g., Dale et al. (2003), Reiter et al. (2005), Green (2006), Turner et al. (2009)). Unfortunately, it is often difficult to adapt them across different domains as they rely mostly on handcrafted components.

In this paper we present a data-driven approach to concept-to-text generation that is domain-independent, conceptually simple, and flexible. Our generator learns from a set of database records and textual descriptions (for some of them). An example from the air travel domain is shown in Figure 1. Here, the records provide a structured representation of the flight details (e.g., departure and arrival time, location), and the text renders some of this information in natural language. Given such input, our model determines which records to talk about (*content selection*) and which words to use for describing them (*surface realization*). Rather than breaking up the generation process into a sequence of local decisions, we perform both tasks jointly. A key insight in our approach is to reduce content selection and surface realization into a common parsing problem. Specifically, we define a probabilistic context-free grammar (PCFG) that captures the structure of the database and its correspondence to natural language. This grammar represents multiple derivations which we encode compactly using a weighted hypergraph (or packed forest), a data structure that defines a weight for each tree.

Following a generative approach, we could first learn the weights of the PCFG by maximising the joint likelihood of the model and then perform generation by finding the best derivation tree in the hypergraph. The performance of this baseline system could be potentially further improved using discriminative reranking (Collins, 2000). Typically, this method first creates a list of n-best candidates from a generative model, and then reranks them with arbitrary features (both local and global) that are either not computable or intractable to compute within the

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 369–378,
Jeju, Republic of Korea, 8-14 July 2012.

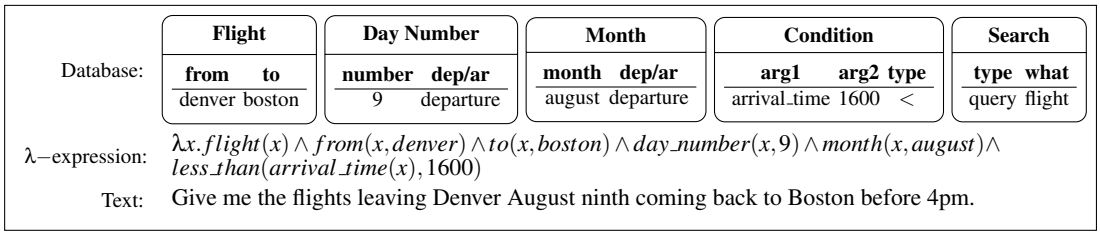

Figure 1: Example of non-linguistic input as a structured database and logical form and its corresponding text. We omit record fields that have no value, for the sake of brevity.

baseline system.

An appealing alternative is to rerank the hypergraph *directly* (Huang, 2008). As it compactly encodes exponentially many derivations, we can explore a much larger hypothesis space than would have been possible with an n-best list. Importantly, in this framework non-local features are computed at all internal hypergraph nodes, allowing the decoder to take advantage of them continuously at all stages of the generation process. We incorporate features that are local with respect to a span of a sub-derivation in the packed forest; we also (approximately) include features that arbitrarily exceed span boundaries, thus capturing more global knowledge. Experimental results on the ATIS domain (Dahl et al., 1994) demonstrate that our model outperforms a baseline based on the best derivation and a state-of-the-art discriminative system (Angeli et al., 2010) by a wide margin.

Our contributions in this paper are threefold: we recast concept-to-text generation in a probabilistic parsing framework that allows to jointly optimize content selection and surface realization; we represent parse derivations compactly using hypergraphs and illustrate the use of an algorithm for *generating* (rather than parsing) in this framework; finally, the application of discriminative reranking to concept-to-text generation is novel to our knowledge and as our experiments show beneficial.

2 Related Work

Early discriminative approaches to text generation were introduced in spoken dialogue systems, and usually tackled content selection and surface realization separately. Ratnaparkhi (2002) conceptualized surface realization (from a fixed meaning representation) as a classification task. Local and non-local information (e.g., word n-grams, long-range dependencies) was taken into account with the use of features in a maximum entropy probability model. More recently, Wong and Mooney (2007) describe an approach to surface realization based on synchronous context-free grammars. The latter are learned using a log-linear model with minimum error rate training (Och, 2003).

Angeli et al. (2010) were the first to propose a unified approach to content selection and surface realization. Their model operates over automatically induced alignments of words to database records (Liang et al., 2009) and decomposes into a sequence of discriminative local decisions. They first determine which records in the database to talk about, then which fields of those records to mention, and finally which words to use to describe the chosen fields. Each of these decisions is implemented as a log-linear model with features learned from training data. Their surface realization component performs decisions based on templates that are automatically extracted and smoothed with domain-specific knowledge in order to guarantee fluent output.

Discriminative reranking has been employed in many NLP tasks such as syntactic parsing (Charniak and Johnson, 2005; Huang, 2008), machine translation (Shen et al., 2004; Li and Khudanpur, 2009) and semantic parsing (Ge and Mooney, 2006). Our model is closest to Huang (2008) who also performs forest reranking on a hypergraph, using both local and non-local features, whose weights are tuned with the averaged perceptron algorithm (Collins, 2002). We adapt forest reranking to generation and introduce several task-specific features that boost performance. Although conceptually related to Angeli et al. (2010), our model optimizes content selection and surface realization simultaneously, rather than as a sequence. The discriminative aspect of two models is also fundamentally different. We have a single reranking component that applies

throughout, whereas they train different discriminative models for each local decision.

3 Problem Formulation

We assume our generator takes as input a set of database records **d** and produces text **w** that verbalizes some of these records. Each record $r \in \mathbf{d}$ has a type $r.t$ and a set of fields f associated with it. Fields have different values $f.v$ and types $f.t$ (i.e., integer or categorical). For example, in Figure 1, *flight* is a record type with fields *from* and *to*. The values of these fields are `denver` and `boston` and their type is categorical.

During training, our algorithm is given a corpus consisting of several *scenarios*, i.e., database records paired with texts like those shown in Figure 1. The database (and accompanying texts) are next converted into a PCFG whose weights are learned from training data. PCFG derivations are represented as a weighted directed hypergraph (Gallo et al., 1993). The weights on the hyperarcs are defined by a variety of feature functions, which we learn via a discriminative online update algorithm. During testing, we are given a set of database records without the corresponding text. Using the learned feature weights, we compile a hypergraph specific to this test input and decode it approximately (Huang, 2008). The hypergraph representation allows us to decompose the feature functions and compute them piecemeal at each hyperarc (or sub-derivation), rather than at the root node as in conventional n-best list reranking. Note that the algorithm does not separate content selection from surface realization, both subtasks are optimized jointly through the probabilistic parsing formulation.

3.1 Grammar Definition

We capture the structure of the database with a number of CFG rewrite rules, in a similar way to how Liang et al. (2009) define Markov chains in their hierarchical model. These rules are purely syntactic (describing the intuitive relationship between records, records and fields, fields and corresponding words), and could apply to any database with similar structure irrespectively of the semantics of the domain.

Our grammar is defined in Table 1 (rules (1)–(9)). Rule weights are governed by an underlying multinomial distribution and are shown in square brackets. Non-terminal symbols are in capitals and denote intermediate states; the terminal symbol α corresponds to all words seen in the training set, and $\mathrm{g}(f.v)$ is a function for generating integer numbers given the value of a field f. All non-terminals, save the start symbol S, have one or more constraints (shown in parentheses), similar to number and gender agreement constraints in augmented syntactic rules.

1. $\mathrm{S} \rightarrow \mathrm{R}(start)$	$[Pr = 1]$
2. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)\ \mathrm{R}(r_j.t)$	$[P(r_j.t \mid r_i.t) \cdot \lambda]$
3. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)$	$[P(r_j.t \mid r_i.t) \cdot \lambda]$
4. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)\ \mathrm{FS}(r, r.f_j)$	$[P(f_j \mid f_i)]$
5. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)$	$[P(f_j \mid f_i)]$
6. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)\ \mathrm{F}(r, r.f)$	$[P(w \mid w_{-1}, r, r.f)]$
7. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)$	$[P(w \mid w_{-1}, r, r.f)]$
8. $\mathrm{W}(r, r.f) \rightarrow \alpha$	$[P(\alpha \mid r, r.f, f.t, f.v)]$
9. $\mathrm{W}(r, r.f) \rightarrow \mathrm{g}(f.v)$	$[P(\mathrm{g}(f.v).mode \mid r, r.f, f.t = int)]$

Table 1: Grammar rules and their weights shown in square brackets.

Rule (1) denotes the expansion from the start symbol S to record R, which has the special *start* type (hence the notation $\mathrm{R}(start)$). Rule (2) defines a chain between two consecutive records r_i and r_j. Here, $\mathrm{FS}(r_j, start)$ represents the set of fields of the target r_j, following the source record $\mathrm{R}(r_i)$. For example, the rule $\mathrm{R}(search_1.t) \rightarrow \mathrm{FS}(flight_1, start)\mathrm{R}(flight_1.t)$ can be interpreted as follows. Given that we have talked about $search_1$, we will next talk about $flight_1$ and thus emit its corresponding fields. $\mathrm{R}(flight_1.t)$ is a non-terminal place-holder for the continuation of the chain of records, and *start* in FS is a special boundary field between consecutive records. The weight of this rule is the bigram probability of two records conditioned on their type, multiplied with a normalization factor λ. We have also defined a *null* record type i.e., a record that has no fields and acts as a smoother for words that may not correspond to a particular record. Rule (3) is simply an escape rule, so that the parsing process (on the record level) can finish.

Rule (4) is the equivalent of rule (2) at the field

level, i.e., it describes the chaining of two consecutive fields f_i and f_j. Non-terminal $\mathrm{F}(r, r.f)$ refers to field f of record r. For example, the rule $\mathrm{FS}(flight_1, from) \rightarrow \mathrm{F}(flight_1, to)\mathrm{FS}(flight_1, to)$, specifies that we should talk about the field *to* of record $flight_1$, after talking about the field *from*. Analogously to the record level, we have also included a special *null* field type for the emission of words that do not correspond to a specific record field. Rule (6) defines the expansion of field F to a sequence of (binarized) words W, with a weight equal to the bigram probability of the current word given the previous word, the current record, and field.

Rules (8) and (9) define the emission of words and integer numbers from W, given a field type and its value. Rule (8) emits a single word from the vocabulary of the training set. Its weight defines a multinomial distribution over all seen words, for every value of field f, given that the field type is categorical or the special *null* field. Rule (9) is identical but for fields whose type is integer. Function $\mathrm{g}(f.v)$ generates an integer number given the field value, using either of the following six ways (Liang et al., 2009): identical to the field value, rounding up or rounding down to a multiple of 5, rounding off to the closest multiple of 5 and finally adding or subtracting some unexplained noise.[1] The weight is a multinomial over the six generation function *modes*, given the record field f.

The CFG in Table 1 will produce many derivations for a given input (i.e., a set of database records) which we represent compactly using a hypergraph or a packed forest (Klein and Manning, 2001; Huang, 2008). Simplified examples of this representation are shown in Figure 2.

3.2 Hypergraph Reranking

For our generation task, we are given a set of database records $\mathbf{d}$, and our goal is to find the best corresponding text $\mathbf{w}$. This corresponds to the best grammar derivation among a set of candidate derivations represented *implicitly* in the hypergraph structure. As shown in Table 1, the mapping from $\mathbf{d}$ to $\mathbf{w}$ is unknown. Therefore, all the intermediate multinomial distributions, described in the previous section, define a hidden correspondence structure $\mathbf{h}$, between records, fields, and their values. We find the best scoring derivation $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$ by maximizing over configurations of $\mathbf{h}$:

$$(\hat{\mathbf{w}}, \hat{\mathbf{h}}) = \arg\max_{w,h} \alpha \cdot \Phi(\mathbf{d}, \mathbf{w}, \mathbf{h})$$

We define the score of $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$ as the dot product between a high dimensional feature representation $\Phi = (\Phi_1, \ldots, \Phi_m)$ and a weight vector α.

We estimate the weights α using the averaged structured perceptron algorithm (Collins, 2002), which is well known for its speed and good performance in similar large-parameter NLP tasks (Liang et al., 2006; Huang, 2008). As shown in Algorithm 1, the perceptron makes several passes over the training scenarios, and in each iteration it computes the best scoring $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$ among the candidate derivations, given the current weights α. In line 6, the algorithm updates α with the difference (if any) between the feature representations of the best scoring derivation $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$ and the the *oracle derivation* $(\mathbf{w}^*, \mathbf{h}^+)$. Here, $\hat{\mathbf{w}}$ is the estimated text, $\mathbf{w}^*$ the gold-standard text, $\hat{\mathbf{h}}$ is the estimated latent configuration of the model and $\mathbf{h}^+$ the oracle latent configuration. The final weight vector α is the average of weight vectors over T iterations and N scenarios. This averaging procedure avoids overfitting and produces more stable results (Collins, 2002).

Algorithm 1: Averaged Structured Perceptron

Input: Training scenarios: $(\mathbf{d_i}, \mathbf{w}^*, \mathbf{h_i^+})_{i=1}^N$

1 $\alpha \leftarrow 0$
2 **for** $t \leftarrow 1 \ldots T$ **do**
3 **for** $i \leftarrow 1 \ldots N$ **do**
4 $(\hat{\mathbf{w}}, \hat{\mathbf{h}}) = \arg\max_{w,h} \alpha \cdot \Phi(\mathbf{d_i}, \mathbf{w_i}, \mathbf{h_i})$
5 **if** $(\mathbf{w_i^*}, \mathbf{h_i^+}) \neq (\hat{\mathbf{w}}_\mathbf{i}, \hat{\mathbf{h}}_\mathbf{i})$ **then**
6 $\alpha \leftarrow \alpha + \Phi(\mathbf{d_i}, \mathbf{w_i^*}, \mathbf{h_i^+}) - \Phi(\mathbf{d_i}, \hat{\mathbf{w}}_\mathbf{i}, \hat{\mathbf{h}}_\mathbf{i})$
7 **return** $\frac{1}{T}\sum_{t=1}^{T}\frac{1}{N}\sum_{i=1}^{N}\alpha_t^i$

In the following, we first explain how we decode in this framework, i.e., find the best scoring derivation (Section 3.3) and discuss our definition for the oracle derivation $(\mathbf{w}^*, \mathbf{h}^+)$ (Section 3.4). Our features are described in Section 4.2.

3.3 Hypergraph Decoding

Following Huang (2008), we also distinguish features into local, i.e., those that can be computed within the confines of a single hyperedge, and non-local, i.e., those that require the prior visit of nodes other than their antecedents. For example, the

[1] The noise is modeled as a geometric distribution.

Alignment feature in Figure 2(a) is local, and thus can be computed a priori, but the **Word Trigrams** is not; in Figure 2(b) words in parentheses are sub-generations created so far at each word node; their combination gives rise to the trigrams serving as input to the feature. However, this combination may not take place at their immediate ancestors, since these may not be adjacent nodes in the hypergraph. According to the grammar in Table 1, there is no direct hyperedge between nodes representing words (W) and nodes representing the set of fields these correspond to (FS); rather, W and FS are connected implicitly via individual fields (F). Note, that in order to estimate the trigram feature at the FS node, we need to carry word information in the derivations of its antecedents, as we go bottom-up.[2]

Given these two types of features, we can then adapt Huang's (2008) approximate decoding algorithm to find $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$. Essentially, we perform bottom-up Viterbi search, visiting the nodes in reverse topological order, and keeping the k-best derivations for each. The score of each derivation is a linear combination of local and non-local features weights. In machine translation, a decoder that implements forest rescoring (Huang and Chiang, 2007) uses the language model as an external criterion of the goodness of sub-translations on account of their grammaticality. Analogously here, non-local features influence the selection of the best combinations, by introducing knowledge that exceeds the confines of the node under consideration and thus depend on the sub-derivations generated so far. (e.g., word trigrams spanning a field node rely on evidence from antecedent nodes that may be arbitrarily deeper than the field's immediate children).

Our treatment of leaf nodes (see rules (8) and (9)) differs from the way these are usually handled in parsing. Since in generation we must emit rather than observe the words, for each leaf node we therefore output the k-best words according to the learned weights α of the **Alignment** feature (see Section 4.2), and continue building our sub-generations bottom-up. This generation task is far from trivial: the search space on the word level is the size of the vocabulary and each field of a record can potentially generate all words. Also, note that in decoding it is useful to have a way to score different output lengths $|\mathbf{w}|$. Rather than setting $\mathbf{w}$ to a fixed length, we rely on a linear regression predictor that uses the counts of each record type per scenario as features and is able to produce variable length texts.

[2]We also store field information to compute structural features, described in Section 4.2.

3.4 Oracle Derivation

So far we have remained agnostic with respect to the oracle derivation $(\mathbf{w}^*, \mathbf{h}^+)$. In other NLP tasks such as syntactic parsing, there is a gold-standard parse, that can be used as the oracle. In our generation setting, such information is not available. We do not have the gold-standard alignment between the database records and the text that verbalizes them. Instead, we approximate it using the existing decoder to find the best latent configuration $\mathbf{h}^+$ given the observed words in the training text $\mathbf{w}^*$.[3] This is similar in spirit to the generative alignment model of Liang et al. (2009).

4 Experimental Design

In this section we present our experimental setup for assessing the performance of our model. We give details on our dataset, model parameters and features, the approaches used for comparison, and explain how system output was evaluated.

4.1 Dataset

We conducted our experiments on the Air Travel Information System (ATIS) dataset (Dahl et al., 1994) which consists of transcriptions of spontaneous utterances of users interacting with a hypothetical online flight booking system. The dataset was originally created for the development of spoken language systems and is partitioned in individual user turns (e.g., *flights from orlando to milwaukee*, *show flights from orlando to milwaukee leaving after six o'clock*) each accompanied with an SQL query to a booking system and the results of this query. These utterances are typically short expressing a specific communicative goal (e.g., a question about the origin of a flight or its time of arrival). This inevitably results in small scenarios with a few words that often unambiguously correspond to a single record. To avoid training our model on a somewhat trivial corpus, we used the dataset introduced in Zettlemoyer

[3]In machine translation, Huang (2008) provides a soft algorithm that finds the forest oracle, i.e., the parse among the reranked candidates with the highest Parseval F-score. However, it still relies on the gold-standard reference translation.

and Collins (2007) instead, which combines the utterances of a single user in one scenario and contains 5,426 scenarios in total; each scenario corresponds to a (manually annotated) formal meaning representation (λ-expression) and its translation in natural language.

Lambda expressions were automatically converted into records, fields and values following the conventions adopted in Liang et al. (2009).[4] Given a lambda expression like the one shown in Figure 1, we first create a record for each variable and constant (e.g., *x*, `9`, `august`). We then assign record types according to the corresponding class types (e.g., variable *x* has class type *flight*). Next, fields and values are added from predicates with two arguments with the class type of the first argument matching that of the record type. The name of the predicate denotes the field, and the second argument denotes the value. We also defined special record types, such as *condition* and *search*. The latter is introduced for every lambda operator and assigned the categorical field *what* with the value `flight` which refers to the record type of variable *x*.

Contrary to datasets used in previous generation studies (e.g., ROBOCUP (Chen and Mooney, 2008) and WEATHERGOV (Liang et al., 2009)), ATIS has a much richer vocabulary (927 words); each scenario corresponds to a single sentence (average length is 11.2 words) with 2.65 out of 19 record types mentioned on average. Following Zettlemoyer and Collins (2007), we trained on 4,962 scenarios and tested on ATIS NOV93 which contains 448 examples.

4.2 Features

Broadly speaking, we defined two types of features, namely lexical and structural ones. In addition, we used a generatively trained PCFG as a baseline feature and an alignment feature based on the co-occurrence of records (or fields) with words.

Baseline Feature This is the log score of a generative decoder trained on the PCFG from Table 1. We converted the grammar into a hypergraph, and learned its probability distributions using a dynamic program similar to the inside-outside algorithm (Li and Eisner, 2009). Decoding was performed approximately via cube pruning (Chiang, 2007), by integrating a trigram language model extracted from the training set (see Konstas and Lapata (2012) for details). Intuitively, the feature refers to the overall goodness of a specific derivation, applied locally in every hyperedge.

Alignment Features Instances of this feature family refer to the count of each PCFG rule from Table 1. For example, the number of times rule $\mathrm{R}(search_1.t) \rightarrow \mathrm{FS}(flight_1, start)\mathrm{R}(flight_1.t)$ is included in a derivation (see Figure 2(a))

Lexical Features These features encourage grammatical coherence and inform lexical selection over and above the limited horizon of the language model captured by Rules (6)–(9). They also tackle anomalies in the generated output, due to the ergodicity of the CFG rules at the record and field level:

Word Bigrams/Trigrams This is a group of non-local feature functions that count word *n*-grams at every level in the hypergraph (see Figure 2(b)). The integration of words in the sub-derivations is adapted from Chiang (2007).

Number of Words per Field This feature function counts the number of words for every field, aiming to capture compound proper nouns and multi-word expressions, e.g., fields *from* and *to* frequently correspond to two or three words such as '*new york*' and '*salt lake city*' (see Figure 2(d)).

Consecutive Word/Bigram/Trigram This feature family targets adjacent repetitions of the same word, bigram or trigram, e.g., '*show me the show me the flights*'.

Structural Features Features in this category target primarily content selection and influence appropriate choice at the field level:

Field bigrams/trigrams Analogously to the lexical features mentioned above, we introduce a series of non-local features that capture field *n*-grams, given a specific record. For example the record *flight* in the air travel domain typically has the values <*from to*> (see Figure 2(c)). The integration of fields in sub-derivations is implemented in fashion similar to the integration of words.

Number of Fields per Record This feature family is a coarser version of the *Field bigrams/trigrams*

[4] The resulting dataset and a technical report describing the mapping procedure in detail are available from `http://homepages.inf.ed.ac.uk/s0793019/index.php?page=resources`

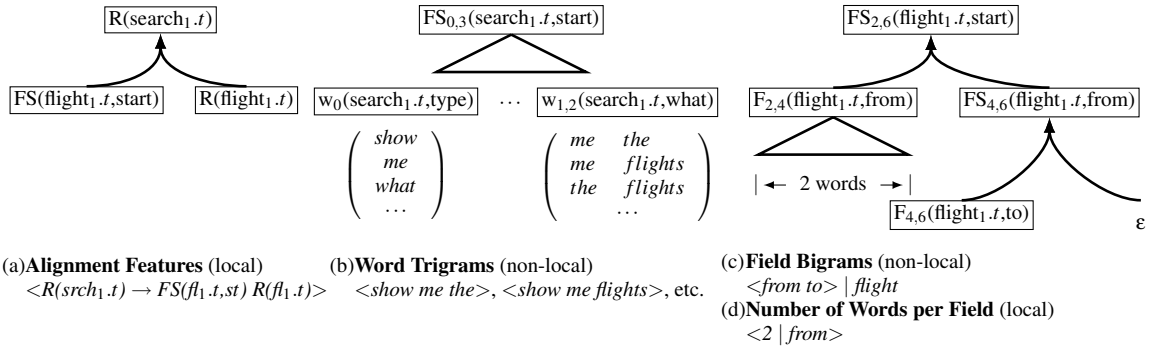

Figure 2: Simplified hypergraph examples with corresponding local and non-local features.

feature, which is deemed to be sparse for rarely-seen records.

Field with No Value Although records in the ATIS database schema have many fields, only a few are assigned a value in any given scenario. For example, the *flight* record has 13 fields, of which only 1.7 (on average) have a value. Practically, in a generative model this kind of sparsity would result in very low field recall. We thus include an identity feature function that explicitly counts whether a particular field has a value.

4.3 Evaluation

We evaluated three configurations of our model. A system that only uses the top scoring derivation in each sub-generation and incorporates only the baseline and alignment features (1-BEST+BASE+ALIGN). Our second system considers the k-best derivations and additionally includes lexical features (k-BEST+BASE+ALIGN+LEX). The number of k-best derivations was set to 40 and estimated experimentally on held-out data. And finally, our third system includes the full feature set (k-BEST+BASE+ALIGN+LEX+STR). Note, that the second and third system incorporate non-local features, hence the use of k-best derivation lists.[5] We compared our model to Angeli et al. (2010) whose approach is closest to ours.[6]

We evaluated system output automatically, using the BLEU-4 modified precision score (Papineni et al., 2002) with the human-written text as reference. We also report results with the METEOR score (Banerjee and Lavie, 2005), which takes into account word re-ordering and has been shown to correlate better with human judgments at the sentence level. In addition, we evaluated the generated text by eliciting human judgments. Participants were presented with a scenario and its corresponding verbalization (see Figure 3) and were asked to rate the latter along two dimensions: fluency (is the text grammatical and overall understandable?) and semantic correctness (does the meaning conveyed by the text correspond to the database input?). The subjects used a five point rating scale where a high number indicates better performance. We randomly selected 12 documents from the test set and generated output with two of our models (1-BEST+BASE+ALIGN and k-BEST+BASE+ALIGN+LEX+STR) and Angeli et al.'s (2010) model. We also included the original text (HUMAN) as a gold standard. We thus obtained ratings for 48 (12×4) scenario-text pairs. The study was conducted over the Internet, using Amazon Mechanical Turk, and was completed by 51 volunteers, all self reported native English speakers.

5 Results

Table 2 summarizes our results. As can be seen, inclusion of lexical features gives our decoder an absolute increase of 6.73% in BLEU over the 1-BEST system. It also outperforms the discriminative system of Angeli et al. (2010). Our lexical features seem more robust compared to their templates. This is especially the case with infrequent records, where their system struggles to learn any meaningful information. Addition of the structural features further boosts performance. Our model increases by 8.69%

[5] Since the addition of these features, essentially incurs reranking, it follows that the systems would exhibit the exact same performance as the baseline system with 1-best lists.

[6] We are grateful to Gabor Angeli for providing us with the code of his system.

System	BLEU	METEOR
1-BEST+BASE+ALIGN	21.93	34.01
k-BEST+BASE+ALIGN+LEX	28.66	45.18
k-BEST+BASE+ALIGN+LEX+STR	30.62	46.07
ANGELI	26.77	42.41

Table 2: BLEU-4 and METEOR results on ATIS.

System	Fluency	SemCor
1-BEST+BASE+ALIGN	2.70	3.05
k-BEST+BASE+ALIGN+LEX+STR	4.02	4.04
ANGELI	3.74	3.17
HUMAN	4.18	4.02

Table 3: Mean ratings for fluency and semantic correctness (SemCor) on system output elicited by humans.

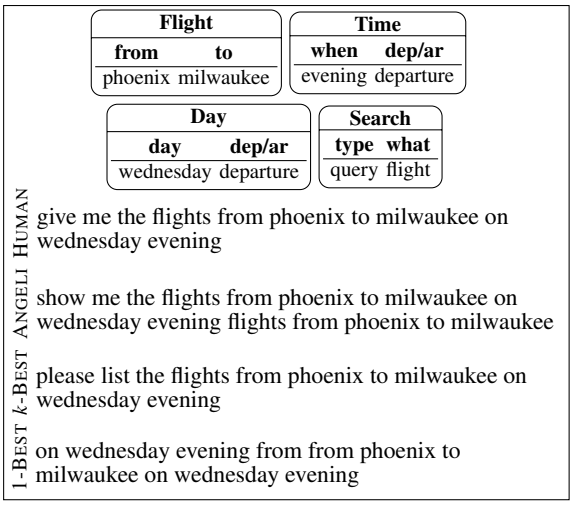

Figure 3: Example of scenario input and system output.

over the 1-BEST system and 3.85% over ANGELI in terms of BLEU. We observe a similar trend when evaluating system output with METEOR. Differences in magnitude are larger with the latter metric.

The results of our human evaluation study are shown in Table 5. We carried out an Analysis of Variance (ANOVA) to examine the effect of system type (1-BEST, k-BEST, ANGELI, and HUMAN) on the fluency and semantic correctness ratings. Means differences were compared using a post-hoc Tukey test. The k-BEST system is significantly better than the 1-BEST and ANGELI ($a < 0.01$) both in terms of fluency and semantic correctness. ANGELI is significantly better than 1-BEST with regard to fluency ($a < 0.05$) but not semantic correctness. There is no statistically significant difference between the k-BEST output and the original sentences (HUMAN).

Examples of system output are shown in Table 3. They broadly convey similar meaning with the gold-standard; ANGELI exhibits some long-range repetition, probably due to re-iteration of the same record patterns. We tackle this issue with the inclusion of non-local structural features. The 1-BEST system has some grammaticality issues, which we avoid by defining features over lexical n-grams and repeated words. It is worth noting that both our system and ANGELI produce output that is semantically compatible with but lexically different from the gold-standard (compare *please list the flights* and *show me the flights* against *give me the flights*). This is expected given the size of the vocabulary, but raises concerns regarding the use of automatic metrics for the evaluation of generation output.

6 Conclusions

We presented a discriminative reranking framework for an end-to-end generation system that performs both content selection and surface realization. Central to our approach is the encoding of generation as a parsing problem. We reformulate the input (a set of database records and text describing some of them) as a PCFG and convert it to a hypergraph. We find the best scoring derivation via forest reranking using both local and non-local features, that we train using the perceptron algorithm. Experimental evaluation on the ATIS dataset shows that our model attains significantly higher fluency and semantic correctness than any of the comparison systems. The current model can be easily extended to incorporate, additional, more elaborate features. Likewise, it can port to other domains with similar database structure without modification, such as WEATHERGOV and ROBOCUP. Finally, distributed training strategies have been developed for the perceptron algorithm (McDonald et al., 2010), which would allow our generator to scale to even larger datasets.

In the future, we would also like to tackle more challenging domains (e.g., product descriptions) and to enrich our generator with some notion of discourse planning. An interesting question is how to extend the PCFG-based approach advocated here so as to capture discourse-level document structure.

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512, Cambridge, MA.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan, June.

David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of International Conference on Machine Learning*, pages 128–135, Helsinki, Finland.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the 17th International Conference on Machine Learning*, pages 175–182, Stanford, California.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8, Philadelphia, Pennsylvania.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: the ATIS-3 corpus. In *Proceedings of the Workshop on Human Language Technology*, pages 43–48, Plainsboro, New Jersey.

Robert Dale, Sabine Geldof, and Jean-Philippe Prost. 2003. Coral: Using natural language generation for navigational assistance. In *Proceedings of the 26th Australasian Computer Science Conference*, pages 35–44, Adelaide, Australia.

Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. 1993. Directed hypergraphs and applications. *Discrete Applied Mathematics*, 42:177–201.

Ruifang Ge and Raymond J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 263–270, Sydney, Australia.

Nancy Green. 2006. Generation of biomedical arguments for lay readers. In *Proceedings of the 5th International Natural Language Generation Conference*, pages 114–121, Sydney, Australia.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08: HLT*, pages 586–594, Columbus, Ohio.

Dan Klein and Christopher D. Manning. 2001. Parsing and hypergraphs. In *Proceedings of the 7th International Workshop on Parsing Technologies*, pages 123–134, Beijing, China.

Ioannis Konstas and Mirella Lapata. 2012. Unsupervised concept-to-text generation with hypergraphs. To appear in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada.

Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 40–51, Suntec, Singapore.

Zhifei Li and Sanjeev Khudanpur. 2009. Forest reranking for machine translation with the perceptron algorithm. In *GALE Book*. GALE.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Sydney, Australia.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore.

Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464, Los Angeles, CA, June. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of*

the 41st Annual Meeting on Association for Computational Linguistics, pages 160–167, Sapporo, Japan.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania.

Adwait Ratnaparkhi. 2002. Trainable approaches to surface natural language generation and their application to conversational dialog systems. *Computer Speech & Language*, 16(3-4):435–455.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge University Press, New York, NY.

Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167:137–169.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *HLT-NAACL 2004: Main Proceedings*, pages 177–184, Boston, Massachusetts.

Ross Turner, Yaji Sripada, and Ehud Reiter. 2009. Generating approximate geographic descriptions. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 42–49, Athens, Greece.

Yuk Wah Wong and Raymond Mooney. 2007. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of the Human Language Technology and the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–179, Rochester, NY.

Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687, Prague, Czech Republic.